AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listings of claims in the application.

1. (currently amended) A method for providing content, comprising the steps of:

receiving a request from a user device for particular content, said request is received at a server;

accessing a mark-up language description of said particular content, said mark-up language description includes one or more source files which describe a behavior of said particular content on a user interface of said user device based on user interactions with the particular content via the user interface, said mark-up language description includes a reference to a view instance element, the view instance element includes a reference to ~~particular content includes~~ data for rendering on said user interface, said ~~one or more source files define~~ mark-up language description defines a connection to an external data source for said data, said external data source is external to said server;

accessing said data at said external data source based on said mark-up language description, ~~one or more source files which define said connection to said external data source~~, said server performs said accessing;

compiling said mark-up language description of said particular content ~~, including said data,~~ to create executable code for said user device, said step of compiling is performed at said server in response to said request, the compiling the mark-up language description includes:

accessing source code for the view instance element, the source code includes a tag name and attributes;

creating a script instruction to call an instantiation view function in a function call;

adding the tag name and attributes to the function call as parameters, the instantiation view function determines, based on the parameters, what kind of objects the user device creates at a runtime of the executable code at the user device; and

compiling the function call to byte code; and

transmitting said executable code and said data from said server to said user device for execution by said user device to allow a user to access ~~provide~~ said particular content and said data

via said user interface ~~according to said one or more source files, and~~ according to said behavior and said user interactions.

2. (cancelled)

3. (cancelled)

4. (previously presented) A method according to claim 1, wherein:

said user device includes a rendering entity and a browser, said rendering entity is a plug-in to said browser, said plug-in is embedded in said browser before said request, and said rendering entity executes said executable code.

5. (currently amended) A method according to claim 1, wherein:

~~after said data is accessed from said external data source, said data is provided in a markup language document,~~ said step of compiling includes parsing the mark-up language description to identify elements of the mark-up language description, translating the elements ~~converting said data in said markup language document~~ to ActionScript and compiling said ActionScript into ActionScript byte code.

6. (cancelled)

7. (previously presented) A method according to claim 1, wherein:

said request for particular content is received from a browser in which a rendering entity is present as a plug-in to said browser, said browser is at said user device, and said rendering entity executes said executable code.

8. (currently amended) A method according to claim 1, wherein: ~~further comprising the steps of:~~

the data comprises ~~accessing~~ media content comprising ~~at least one of audio,~~ video; ~~and a movie, said particular content includes said media content;~~

~~providing a reference in said mark-up language description to a media file which contains said media content, said media file is external to said mark-up language description;~~ and

~~transmitting~~ said media ~~file~~ content is transmitted with said executable code from said server to said user device for use by a rendering entity at said user device in allowing the user to access said video ~~rendering said media content~~ on said user interface when said media content ~~file~~ is referenced when said executable code is executed.

9. (cancelled)

10. (cancelled)

11. (previously presented) A method according to claim 1, further comprising the step of:

authenticating said request, said steps of compiling and transmitting are only performed if said step of authenticating is successful, different types of authenticating are provided for at least one of: a) different types of content and b) each item of content.

12. (cancelled)

13. (currently amended) A method according to claim 1, wherein the request from the user device for said particular content is a request for ~~includes~~ a first application, the first application ~~which~~ runs on said user device ~~after~~ when said executable code is executed at ~~transmitted from said server to~~ said user device, the method further comprising the steps of:

receiving a request at said server from said first application running on the user device for ~~second content when said first application which runs on said user device calls said~~ a second application;

in response to the request, accessing and compiling the second application; ~~a mark-up language description of said second content;~~

~~compiling said mark-up language description of said second content;~~ and

transmitting said compiled ~~mark-up language description of said~~ second application ~~content~~ from said server to said user device.

14.-20. (cancelled)

21. (currently amended) A method for providing content, comprising the steps of:

receiving a request for content that includes data other than code, said data is for rendering on a user interface at a client, and said request is received at a server;

accessing a mark-up language description associated with said content at said server, said mark-up language description defines a connection to an external data source for said data, said external data source is external to said server;

acquiring said data from said external data source in response to said mark-up language description, said data is acquired by said server;

compiling said content at said server to create executable code, said content is based on said mark-up language description and said data, said executable code includes a representation of said data, said step of compiling is performed in response to said request; and

transmitting said executable code from said server to said client, said executable code implements said user interface, said user interface allows a user to access and dynamically interact with said data.

22. (original) A method according to claim 21, wherein:

said request is from said client.

23.-27. (cancelled)

28. (currently amended) One or more processor readable storage devices having processor readable code embodied on said processor readable storage devices, said processor readable code for programming one or more processors to perform a method comprising the steps of:

receiving a request for particular content from a browser, said browser having a plug-in embedded therein, said request is received at a server;

in response to the request, accessing a mark-up language description of said particular content, said mark-up language description references a media file comprising at least one of audio, video and a movie;

compiling said mark-up language description of said particular content to create executable code for said plug-in to said browser, said executable code provides said particular content, said step of compiling is performed at said server in response to said request; and

transmitting said executable code and said media file from said server to said plug-in, said plug-in renders said particular content based on said executable code and said media file.

29. (cancelled)

30. (original) One or more processor readable storage devices according to claim 28, wherein:

said executable code implements a user interface that provides access to said particular content.

31. (original) One or more processor readable storage devices according to claim 28, wherein:

said particular content includes data; and

said data is compiled to executable code during said step of compiling.

32. (cancelled)

33. (currently amended) One or more processor readable storage devices having processor readable code embodied on said processor readable storage devices, said processor readable code for programming one or more processors to perform a method comprising the steps of:

receiving a request for particular content, said request is received at a server from a web client in which a plug-in is embedded;

accessing a mark-up language description of ~~first code associated with~~ said particular content, said mark-up language description includes one or more source files which describe a

behavior of said particular content on a user interface of said web client based on dynamic user interactions with the particular content via the user interface;

in response to said request, compiling said mark-up language description ~~first code~~ to create executable code for said plug-in to said web client, said executable code implements said ~~a~~ user interface, ~~that provides access to said particular content,~~ said step of compiling is performed at said server in response to said request, the compiling the mark-up language description includes:

accessing source code for the view instance element, the source code includes a tag name and attributes;

create a script instruction to call an instantiation view function in a function call;

add the tag name and attributes to the function call as parameters, the instantiation view function determines, based on the parameters, what kind of objects the plug-in creates at a runtime of the executable code at the user device; and

compiling the function call to byte code; and

transmitting said executable code from said server to said plug-in for execution by said plug-in to allow a user to access and dynamically interact with said particular content via said user interface.

34.-40. (cancelled)

41. (currently amended) An apparatus, comprising:

one or more storage devices; and

one or more processors in communication with said one or more storage devices, said one or more processors: (a) receive a request for particular content from an HTTP client having a plug-in embedded therein, said request is received at a server, (b) access a mark-up language description of said particular content, said mark-up language description describes a behavior of said particular content on a user interface of said HTTP client based on user interactions with the particular content via the user interface, said mark-up language description includes a reference to a view instance element, the view instance element includes a reference to data for rendering on said user interface;, and (c) compile said mark-up language description of said particular content to create executable code for said plug-in to said HTTP client, the compiling the mark-up language description includes:

accessing source code for the view instance element, the source code includes a tag name and attributes;

create a script instruction to call an instantiation view function in a function call;

add the tag name and attributes to the function call as parameters, the instantiation view function determines, based on the parameters, what kind of objects the user device creates at a runtime of the executable code at the user device; and

compiling the function call to byte code; and

said executable code provides said particular content, said compiling is performed at said server in response to said request, and said executable code is transmitted from said server to said plug-in for execution by said plug-in to provide said particular content via said user interface according to said behavior and said user interactions.

42. (cancelled)

43. (original) An apparatus according to claim 41, wherein:

said particular content includes data; and

said data is compiled to executable code during said step of compiling.

44. (currently amended) An apparatus according to claim 41, wherein the view instance element includes a reference to media content, and the one or more processors:

access the media content;

create an object representation of the media content, the object representation includes the media content and fields which store attributes of the media content, the attributes include a name of the media content and a format of the media content;

remove the media content from the object representation and insert a reference to the media content into the object representation in place of the media content, then compile the object representation to byte code;

create a tag header;

add the media content, but not the compiled object representation, to the tag header; and

transmit the compiled object representation with the compiled mark-up language description from said server to said user device for execution by said user device to provide said particular content and said data via said user interface according to said behavior and said user interactions ~~said particular content includes at least one of audio, video and a movie.~~

45. (currently amended) An apparatus, comprising:

one or more storage devices; and

one or more processors in communication with said one or more storage devices, said one or more processors perform a method comprising the steps of:

receiving a request for particular content, said request is received at a server, said request is from a client which includes a browser and a rendering engine that is different than said browser but operates in connection with said browser;

accessing first code associated with said particular content at said server, said first code comprises elements that are identified by markup language tags, at least one of said elements references a media content ~~source~~ external to said server;

accessing the media content;

compiling said first code to create executable code for said rendering engine, said step of compiling is performed at said server in response to said request;~~,~~

creating an object representation of the media content, the object representation includes the media content and fields which store attributes of the media content, the attributes include a name of the media content and a format of the media content;

removing the media content from the object representation and inserting a reference to the media content into the object representation in place of the media content, then compiling the object representation to byte code;

creating a tag header;

adding the media content, but not the compiled object representation, to the tag header; and

transmitting said executable code with the compiled object representation from said server to said client for rendering of said particular content and the media content by said

rendering engine, said executable code implements a user interface at said client that provides access to said particular content.

46. (cancelled)

47. (currently amended) An apparatus according to claim 45, wherein the one or more processors implement a media transcoder which includes interfaces for images, audio, graphics and video, and said media transcoder ~~method further comprises the steps of:~~

~~accessing media content, said particular content includes said media content, at least one of said elements identifies said media content;~~

~~transcoding~~ transcodes said media content to an accepted format before the media content is added to the tag header, said transcoding is separate from said compiling of the first code,

~~; and~~

~~transmitting said transcoded media content with said executable code to said client for rendering of said transcoded media content by said rendering engine.~~

48.-50. (cancelled)

51. (previously presented) A method according to claim 21, wherein:

said data is media data comprising at least one of audio, video and a movie.

52. (currently amended) A method according to claim 4, wherein:

the server has separate object code generators and compilers for different types of rendering entities;

said request is received at said server from said user device and includes an indication that identifies a type of said rendering entity from among the different types ~~a group~~ of rendering entities; and

said compiling includes creating said executable code specific for said type of rendering entity using corresponding ones of the object code generators and compilers, in response to said indication.

53. (previously presented) A method according to claim 1, wherein:

said executable code comprises one or more binary files.

54. (previously presented) A method according to claim 1, wherein:

said executable code comprises at least one of object code and byte code.

55. (currently amended) One or more processor readable storage devices according to claim 33, wherein:

said mark-up language description ~~first code~~ comprises elements which are identified by markup language tags.

56. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

at least one of said elements defines a view template of a user interface element, said view template is instantiated when said executable code is executed by said rendering entity.

57. (previously presented) One or more processor readable storage devices according to claim 56, wherein:

said elements comprise at least one element which defines a view class which supplies default properties, behavior, and child views which the view template instantiates, the child views are associated with a parent view.

58. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

at least one of said elements references a media file comprising at least one of audio, video and a movie.

59. (cancelled)

60. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

at least one of said elements references a media file that contains an animation.

61. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

at least one of said elements references a media file that contains a movie.

62. (previously presented) One or more processor readable storage devices according to claim 28, wherein:

said media file comprises a .SWF file, said markup language description references said .SWF file.

63. (cancelled)

64. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

at least one of said elements provides an inline definition of formatted text.

65. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

at least one of said elements provides an inline definition of vector graphics.

66. (cancelled)

67. (previously presented) A method according to claim 1, wherein:

said markup language description comprises elements which are identified by markup language tags; and

said elements comprise at least one element which references said connection to said external data source.

68. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

at least one of said elements defines a connection to a web service which is external to said server.

69. (previously presented) A method according to claim 1, wherein:

said compiling comprises parsing said markup language description to identify first and second types of elements in the markup language description, providing said first type of element to a first compiling module which is appropriate for said first type of element to obtain first object code, providing said second type of element to a second compiling module which is appropriate for said second type of element to obtain second object code, and assembling said first and second object code into a single executable; and

said transmitting said executable code comprises transmitting said single executable to said user device.

70. (previously presented) A method according to claim 69, wherein:

said first type of element provides a script which defines said behavior of said particular content, and said second type of element defines said connection to said external data source.

71. (cancelled)

72. (cancelled)

73. (previously presented) A method according to claim 4, wherein:

said rendering entity is a Flash player.

74. (cancelled)

75. (cancelled)

76. (cancelled)

77. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

said elements comprises elements which define script code, said script code specifies a visual appearance of said user interface.

78. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

said elements comprises elements which define script code, said script code specifies an application logic of said mark-up language description.

79. (previously presented) One or more processor readable storage devices according to claim 55, wherein:

said elements comprises elements which define script code, said script code specifies a connection to an external data source, said external data source includes data for rendering on said user interface by said plug-in.

80. (previously presented) A method according to claim 28, wherein:

said plug-in is a Flash player.

81. (currently amended) A method according to claim 8, further comprising:

providing an object in the executable code which identifies ~~at least one of~~ a name and a format of the media content, the ~~at least one of~~ name and ~~a~~ format are ~~is~~ provided via the user interface when said media content is rendered.

82. (previously presented) A method according to claim 8, wherein:

said request for particular content is received from a browser in which a plug-in to said browser is present, said browser is at said user device, and said plug-in renders said media content.

83. (previously presented) A method according to claim 1, wherein:

said executable code provides a script which is executed when a specified event occurs when a user interacts with the particular content via the user interface, the specified event is based on user control of a pointing device or a key press.

84. (new) A method according to claim 1, wherein:

the executable code, when executed at the user device, creates objects which are displayed on the user interface, the objects are created based on the instantiation view function, the tag name and the attributes.

85. (new) A method according to claim 1, wherein:

the executable code, when executed at the user device, calls a predefined instantiation function associated with the tag name, and passes the attributes to the predefined instantiation function.

86. (new) A method according to claim 1, wherein:

the executable code, when executed at the user device, calls a user-defined instantiation function associated with the tag name, and passes the attributes to the user-defined instantiation function.

87. (new) A method according to claim 1, wherein the data comprises media content, the method further comprising:

receiving the media content at the server, from the external data source;

creating an object representation of the media content, the object representation includes the media content and fields which store attributes of the media content, the attributes include a name of the media content and a format of the media content;

removing the media content from the object representation and inserting a reference to the media content into the object representation in place of the media content, then compiling the object representation to byte code;

creating a tag header;

adding the media content, but not the compiled object representation, to the tag header; and

transmitting the compiled object representation with the compiled mark-up language description from said server to said user device for execution by said user device to provide said particular content and said data via said user interface according to said behavior and said user interactions.

88. (new) A method according to claim 87, further comprising:

assembling the compiled mark-up language description and the compiled object representation into a single executable which is transmitted as said executable code from said server to said user device.

89. (new) A method according to claim 87, further comprising:

transcoding said media content before the adding the media content to the tag header, said transcoding is separate from the compiling of the object, the media content comprises audio which is transcoded between MP3 and WAV.

90. (new) A method according to claim 87, further comprising:

transcoding said media content before the adding the media content to the tag header, said transcoding is separate from the compiling of the object, the media content comprises video which is transcoded between any two of the following formats: MPEG, MPEG2, SORENSON, REAL, and Animated GIF.

91. (new) A method according to claim 87, further comprising:

transcoding said media content before the adding the media content to the tag header, said transcoding is separate from the compiling of the object, the media content comprises an image which is transcoded between any two of the following formats: JPEG, GIF, BMP, and PNG.

92. (new) A method according to claim 87, further comprising:

transcoding said media content before the adding the media content to the tag header, said transcoding is separate from the compiling of the object, the media content comprises graphics which is transcoded between any two of the following formats: SVG, HTML, PDF and SWF.

93. (new) A method according to claim 87, wherein:
the media content comprises audio.

94. (new) A method according to claim 87, wherein:
the media content comprises video.

95. (new) A method according to claim 87, wherein:
the media content comprises a movie.

96. (new) A method according to claim 87, wherein:
the media content comprises an animation.

97. (new) A method according to claim 87, wherein:
said media content comprises a .SWF file.

98. (new) One or more processor readable storage devices according to claim 33, wherein the view instance element includes a reference to media content, and the method performed further comprises:

accessing the media content;

creating an object representation of the media content, the object representation includes the media content and fields which store attributes of the media content, the attributes include a name of the media content and a format of the media content;

removing the media content from the object representation and inserting a reference to the media content into the object representation in place of the media content, then compiling the object representation to byte code;

creating a tag header;

adding the media content, but not the compiled object representation, to the tag header; and

transmitting the compiled object representation with the compiled mark-up language description from said server to said plug-in for execution by said plug-in to provide said particular content and said data via said user interface according to said behavior and said user interactions

99. (new) One or more processor readable storage devices according to claim 98, wherein the method performed further comprises:

assembling the compiled mark-up language description and the compiled object representation into a single executable which is transmitted as said executable code from said server to said plug-in.

100. (new) One or more processor readable storage devices according to claim 98, wherein the method performed further comprises:

transcoding said media content before the adding the media content to the tag header, said transcoding is separate from the compiling of the object.

101. (new) An apparatus according to claim 45, wherein:

the media file comprises at least one of audio, video and a movie.

102. (new) An apparatus according to claim 45, wherein the compiling of the first code comprises:

accessing source code for the at least one of said elements which references the media file, the source code includes a tag name and attributes;

create a script instruction to call an instantiation view function in a function call;

add the tag name and attributes to the function call as parameters, the instantiation view function determines, based on the parameters, what kind of objects the client creates at a runtime of the executable code at the client; and

compiling the function call to byte code.